CHINESE REMAINDER THEOREM

an introduction
OVERVIEW

• Chinese Remainder Theorem
• RSA Decryption
CHINESE REMAINDER THEOREM

• First found in an ancient Chinese puzzle:

There are certain things whose number is unknown. Repeatedly divided by 3, the remainder is 2; by 5 the remainder is 3; and by 7 the remainder is 2. What will be the number?
CHINESE REMAINDER THEOREM

• In modern notation

\[ x = 2 \pmod{3} \]
\[ x = 3 \pmod{5} \]
\[ x = 2 \pmod{7} \]
CHINESE REMAINDER THEOREM

• A solution was given in a poem for the more general problem of

\[ x = a_1 \pmod{3} \]
\[ x = a_2 \pmod{5} \]
\[ x = a_3 \pmod{7} \]
CHINESE REMAINDER THEOREM

三人同行七十里
Three men walking together for seventy miles,
五树梅花二十一枝
Five plum trees with twenty one branches in flower,
七子团圆正月半
Seven disciples gathering right by the half-moon,
一白零五转回起
One hundred and five and we’re back at the start
CHINESE REMAINDER THEOREM

• What does the poem mean?

\[
x = a_1 \pmod{3}
\]
\[
x = a_2 \pmod{5}
\]
\[
x = a_3 \pmod{7}
\]
\[
x = 70a_1 + 21a_2 + 15a_3 \pmod{105}
\]
Why is the solution correct?

\[ x = 70a_1 + 21a_2 + 15a_3 \pmod{105} \]

\[ x = \theta_1 + \theta_2 + \theta_3 \pmod{3} \]

notice that

70 = 1 \pmod{3} = 0 \pmod{5} = 0 \pmod{7}

21 = 0 \pmod{3} = 1 \pmod{5} = 0 \pmod{7}

15 = 0 \pmod{3} = 0 \pmod{5} = 1 \pmod{7}
CHINESE REMAINDER THEOREM

• Solution for the puzzle

There are certain things whose number is unknown. Repeatedly divided by 3, the remainder is 2; by 5 the remainder is 3; and by 7 the remainder is 2. What will be the number?

\[
x = 70(2) + 21(3) + 15(2) \pmod{105}
= 140 + 63 + 30 \pmod{105}
= 233 \pmod{105}
= 23 \pmod{105}
\]
CHINESE REMAINDER THEOREM

- **CRT Algorithm - Gauss**

  If \( x = a_1 \pmod{m_1} = a_2 \pmod{m_2} = \ldots = a_n \pmod{m_n} \)
  where \( m_1, m_2, \ldots, m_n \) are relatively prime to each other, then the solution is
CHINESE REMAINDER THEOREM

• CRT Algorithm - Gauss

where

\[ M_i = \prod_{k=1}^{n} \frac{m_k}{m_i} \]

\[ y_i = \left( M_i \right)^{-1} (\text{mod } m_i) \]

(modulo inverses can be found using Extended Euclidean Algorithm – see any elementary number theory book)
CHINESE REMAINDER THEOREM

• Example

\[ x = 1 \pmod{2} \quad M_1 = 105 \pmod{210} = 1 \pmod{2} \quad y_1 = 1 \]
\[ x = 2 \pmod{3} \quad M_2 = 70 \pmod{210} = 1 \pmod{3} \quad y_2 = 1 \]
\[ x = 3 \pmod{5} \quad M_3 = 42 \pmod{210} = 2 \pmod{5} \quad y_3 = 3 \]
\[ x = 4 \pmod{7} \quad M_4 = 30 \pmod{210} = 2 \pmod{7} \quad y_4 = 4 \]

\[ x = (1)(105)(1) + (2)(70)(1) + (3)(42)(3) + (4)(30)(4) \pmod{210} \]
\[ = 105 + 140 + 378 + 480 \pmod{210} \]
\[ = 53 \pmod{210} \]
CHINESE REMAINDER THEOREM

• Extension of CRT – Gauss

Previously, CRT is limited to cases where the moduli $m_1, m_2, ..., m_n$ are relatively prime to each other. Gauss showed how to convert problems where the moduli are not relatively prime to each other, to the known type mentioned earlier.
CHINESE REMAINDER THEOREM

• Example

\[ x = 4 \pmod{6} \]
\[ x = 2 \pmod{7} \]
\[ x = 6 \pmod{8} \]
\[ x = 7 \pmod{9} \]
\[ x = 142 \pmod{514} \]

• After reduction

\[ x = 0 \pmod{2} \]
\[ x = 1 \pmod{3} \]
\[ x = 2 \pmod{7} \]
\[ x = 6 \pmod{8} = 0 \pmod{2} \]
\[ x = 7 \pmod{9} = 1 \pmod{3} \]
RSA DECRYPTION

• RSA Refresher

Fermat’s little theorem
For any prime \( p \) and any positive integer \( a \). then \( a^{p-1} = 1 \mod p \).

Euler’s theorem (generalization of theorem above)
If \( n \) and \( a \) are positive integers and \( a \) is co-prime to \( n \). then \( a^{\phi(n)} = 1 \mod n \), where \( \phi(n) \) is Euler’s totient function.
RSA DECRYPTION

- **RSA Refresher**

  \[ n = pq, \ p \text{ and } q \text{ are safe primes} \]
  \[ \phi(n) = (p-1)(q-1) \]
  choose a small \( e \), such that it is co-prime to \( \phi(n) \)
  choose \( d = e^{-1} \pmod{\phi(n)} \) so that \( de = 1 \pmod{\phi(n)} \)
RSA DECRYPTION

- **RSA Refresher**

Let message = \( M \) \((M < n)\)

Encrypted message = \( C = M^e \mod n \)

To decrypt find \( C^d \) = \((M^e)^d \mod n\)

\[ = M^{k(\phi(n)) + 1} \mod n \]

\[ = (M^{\phi(n)})^k M \mod n \]

\[ = M \mod n \]
RSA DECRYPTION

• *Problems in practice*

The value $e$ is chosen to be small so that public encryption can be done fast.

However, $d$ would usually be big, making decryption inevitably much slower than encryption.

E.g.: $p = 1289$  \hspace{1cm} q = 997

$n = 1285133$  \hspace{1cm} \varphi(n) = 1282848

$e = 5$  \hspace{1cm} d = 769709
RSA DECRYPTION

• Using CRT to accelerate decryption

Since \( p \) and \( q \) are known by private key owner, we can use CRT to accelerate RSA decryption.

To do that, we need to find out the following

\[
d_p = e^{-1} \pmod{(p - 1)} \quad \rightarrow \quad d_p e = 1 \pmod{(p - 1)}
\]

\[
d_q = e^{-1} \pmod{(q - 1)} \quad \rightarrow \quad d_q e = 1 \pmod{(q - 1)}
\]
RSA DECRYPTION

- Using CRT to accelerate decryption

E.g.: 

\[
\begin{align*}
 p &= 1289 & q &= 997 \\
 p - 1 &= 1288 & q - 1 &= 996 \\
 e &= 5 & d &= 769709 \\
 n &= 1285133 & \phi(n) &= 1282848 \\
 d_p &= 5^{-1} \pmod{1288} = 773 \pmod{1288} \\
 d_q &= 5^{-1} \pmod{996} = 797 \pmod{996}
\end{align*}
\]
RSA DECRYPTION

• *Using CRT to accelerate decryption*

\[ M_p = C^{d_p} = M^{e d_p} \pmod{p} = M^{k_p (p-1)+1} \pmod{p} = M \pmod{p} \]

\[ M_q = C^{d_q} = M^{e d_q} \pmod{q} = M^{k_q (q-1)+1} \pmod{q} = M \pmod{q} \]

After finding \( M_p \) and \( M_q \) we use CRT to find \( M \).
RSA DECRYPTION

• **PKCS**

  One drawback of the classical Gaussian CRT algorithm is the need to do a modular reduction by $M$, which is rather computationally exhaustive.

  Thus, in PKCS #1 v2.1, the classical Gaussian CRT algorithm shown earlier is not used.
RSA DECRYPTION

- **PKCS**
  Instead PKCS #1 v2.1, uses a special case of Garner’s algorithm for two moduli, which only needs to do modular reductions in \( p \).
  In addition to finding \( M_p \) and \( M_q \), we need to find \( q_{inv} = q^{-1}(mod\ p) \) (where \( p>q \))
RSA DECRYPTION

• **PKCS**
  
  An intermediate value $h$ is then calculated
  
  $$h = q_{inv} (M_p - M_q) \mod p$$
  
  this intermediate value is then used to calculate $M$ directly
  
  $$M = M_q + hq$$
RSA DECRYPTION

• Conclusion
  Even though we have to do two exponentiations instead of one and there are additional steps involved in doing CRT, overall, the decryption would be about four times faster.
REFERENCES

END